Printing Best Practices for Your Application

Session 212

These are confidential sessions—please refrain from filming or taking pictures.

Graphics and Media

Printing Best Practices for Your Application

Richard Blanchard

http://developer.apple.com/wwdc2005/

Thank You Again

- Thousands of applications printing beautiful documents
- Thousands of printer models supported
  - Gigabytes of drivers

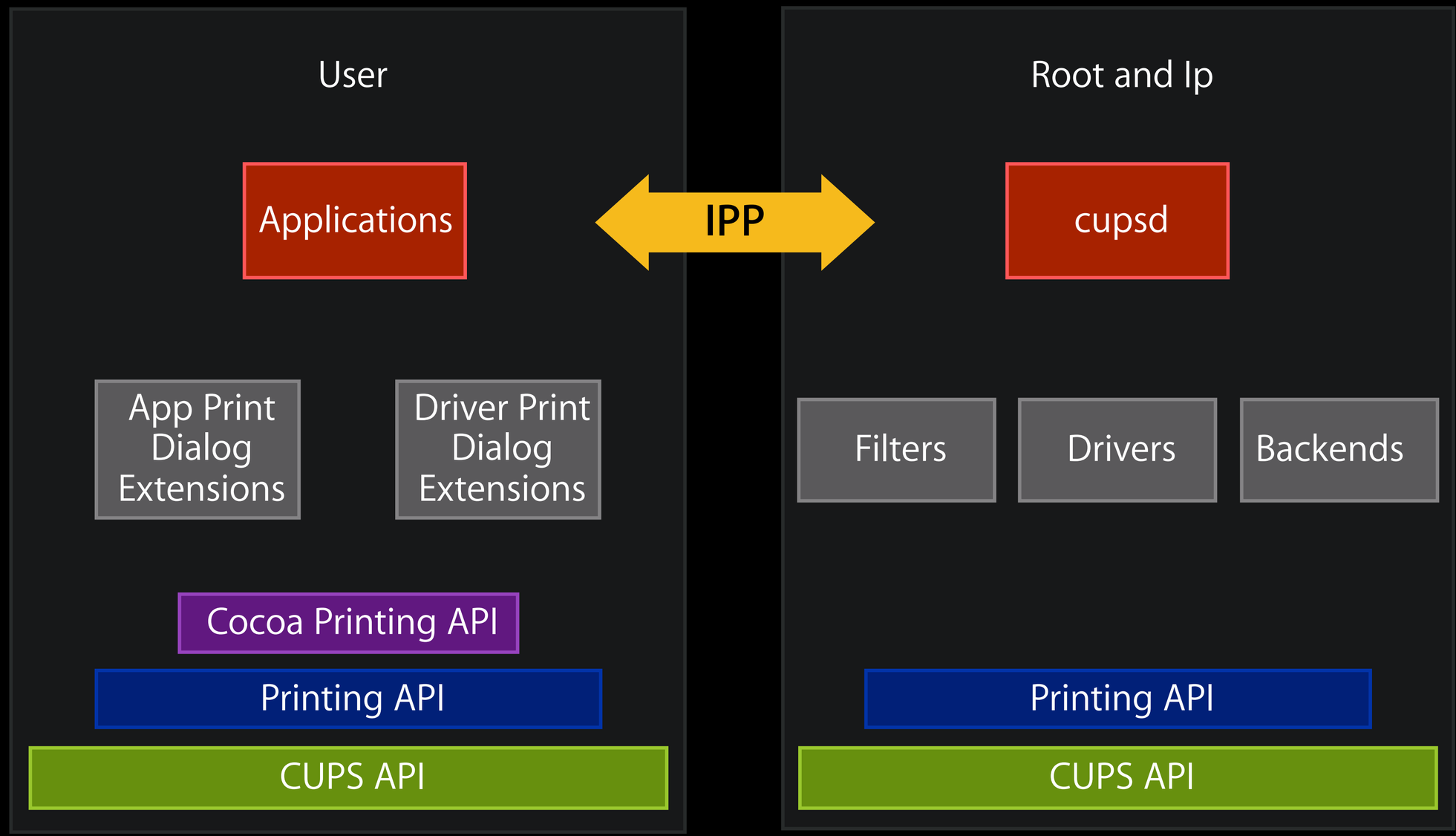
Two Large Sets of Clients

Applications

Print System

Print Drivers

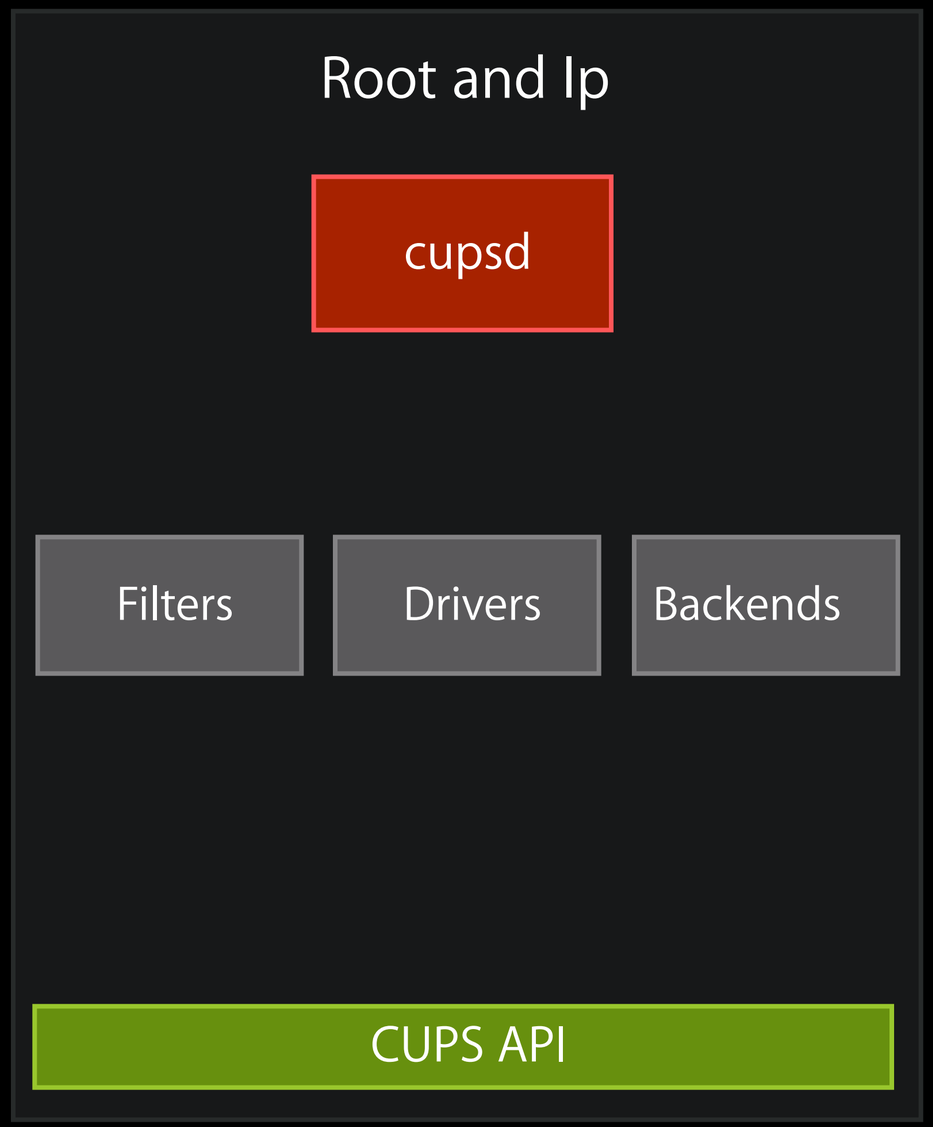
Living in Different Worlds



Overview

- Tour the printing system
  - Stop and point out
    - Developer opportunities
    - New Tiger features

CUPS



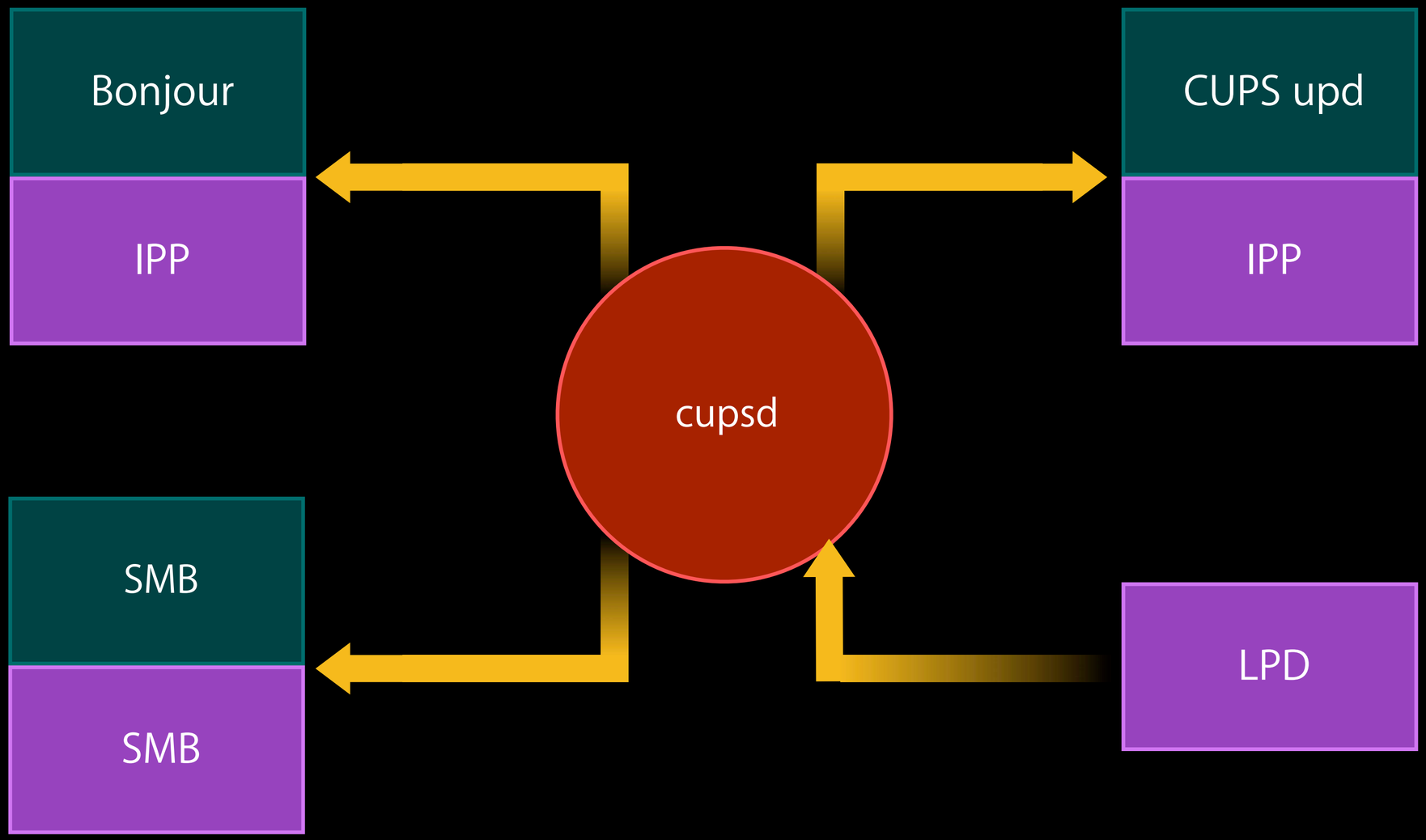
Print Server

- CUPS 1.1.23 in Tiger
  - Bonjour printer publishing
  - Lazy CUPS launching
- CUPS is open source
  - See the Darwin repository
  - cvs to get the latest

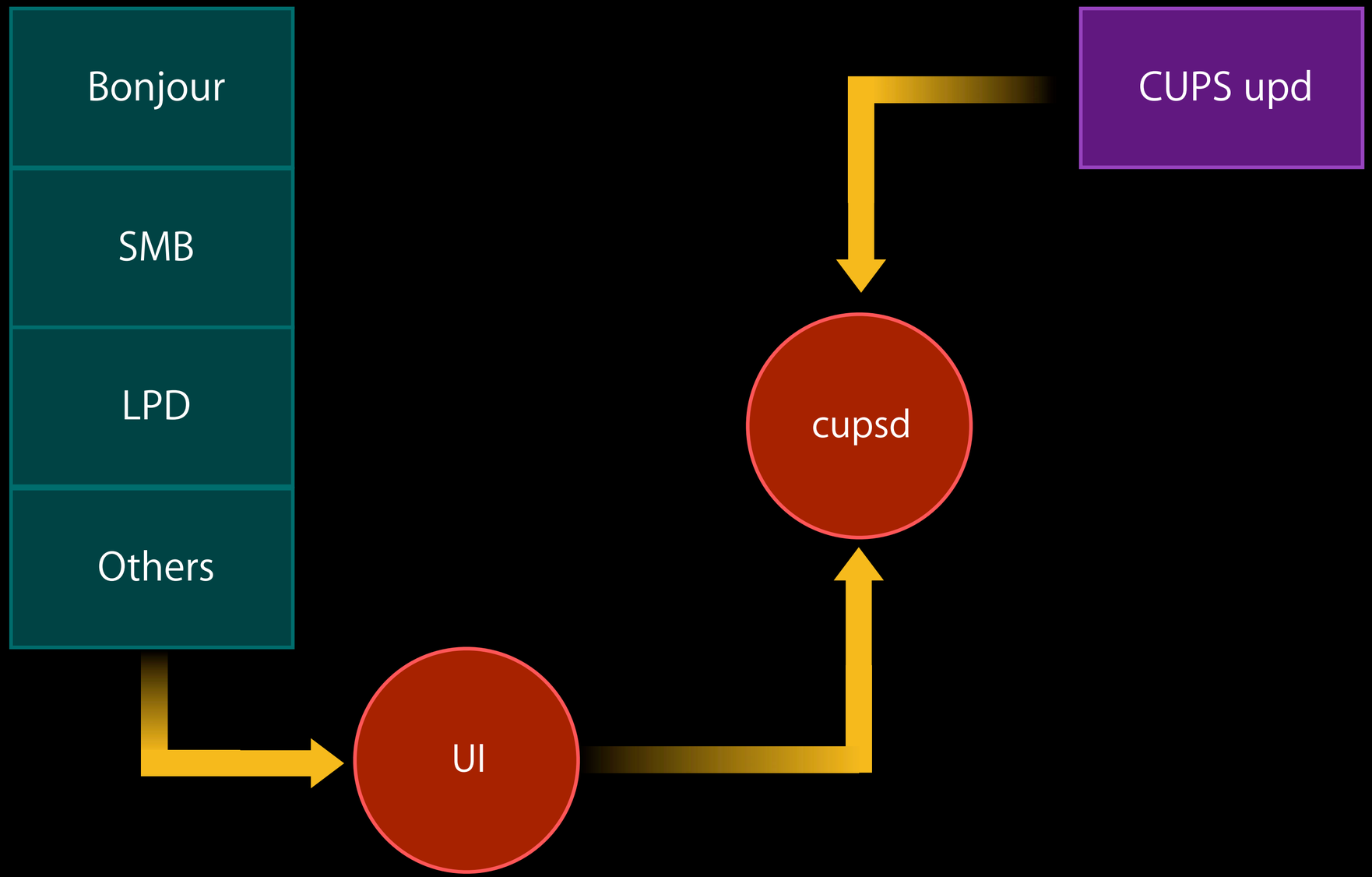
CUPS Functions

- Internet Printing Protocol server
  - Publishes shared printer queues
  - Listens for shared printer queues
  - Web interface
  - Queues print jobs
  - Starts print jobs

Sharing Printers



Finding Shared Printers



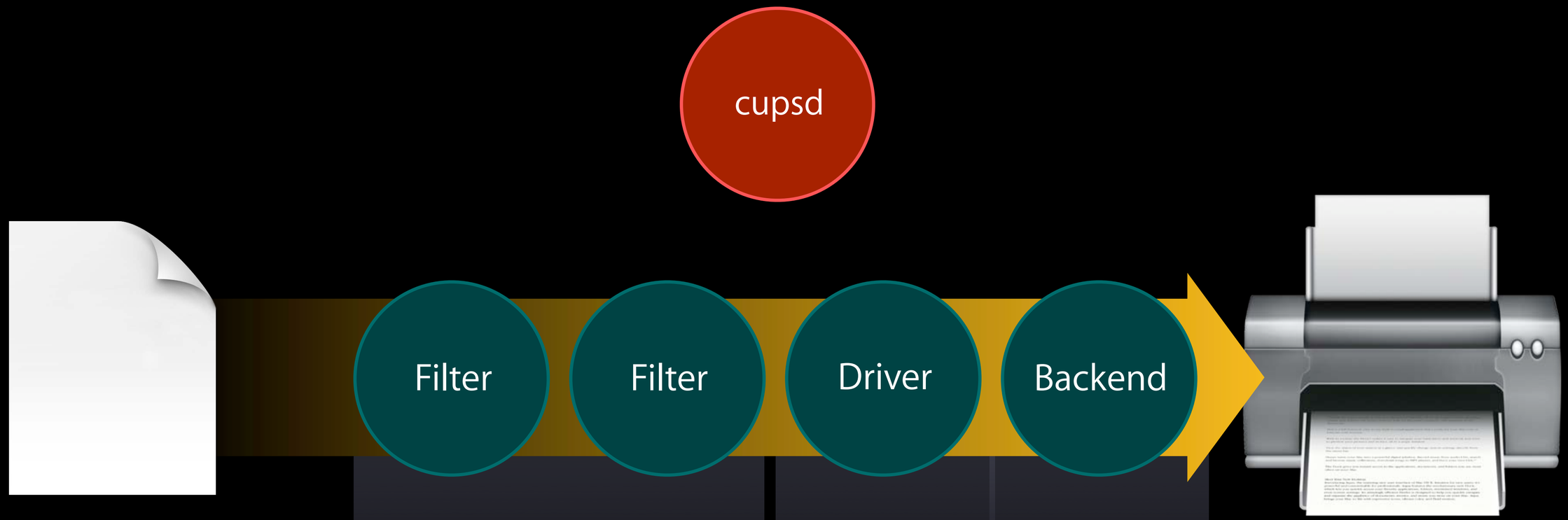
Queuing Jobs

- Print queues support
  - Priorities
  - Scheduling
  - Pools of printers (classes)

Starting a Print Job

- cupsd create a chain of processes to print a job
  - Filters
  - Driver
  - Backend
- stdout of one process hooks to stdin of the next process

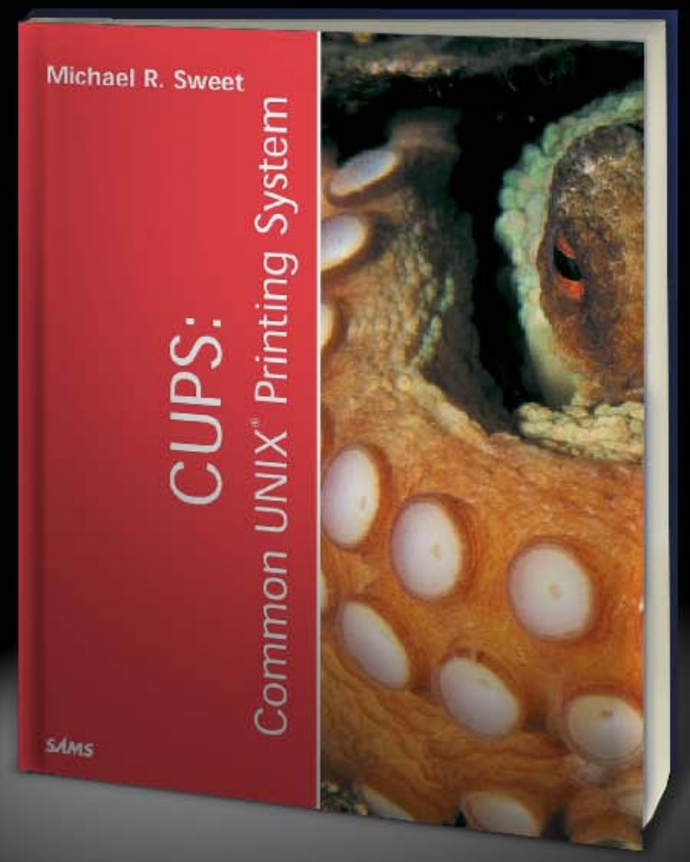
Print Processes



When Things Go Wrong

- Look in /var/log/cups/error_log
- Open /etc/cups/cupsd.conf
- Change ‘LogLevel info’ to
  - LogLevel debug
- Restart print server
  - sudo killall -HUP cupsd

For More Information



Logged Output

```
I [] Started filter /usr/libexec/cups/filter/cgpdftops
(PID 15642) for job 18.
I [] Started filter /usr/libexec/cups/filter/pstops
(PID 15643) for job 18.
I [] Started backend /usr/libexec/cups/backend/lpd
(PID 15644) for job 18.
```

Developer Opportunities

- Mime Filters
  - Independent of driver
- Driver
  - Usually written by printer manufacturer
  - Alternate drivers
- Backend
  - Send the driver output somewhere else

Filters

- Enable the print system to directly accept a specific mime type
  - Drag-and-Drop printing on a printer proxy
  - Direct Printing API

Writing a Filter

- A tool to convert one mime type to another
- Command line arguments
  - Printer job user title copies options [filename]
- Read either from stdin or from the provided file
- Write to stdout
- Errors to stderr

Installing a Filter

- In /etc/cups/ add
  - <vendor>.type and <vendor>.convs files
  - .type describes how the file type can be identified
  - .convs declares the tool and the cost to convert from one mime type to another

Installing Filter Example

See apple.types and apple.convs

```
image/pict  pict pic PICT string(522, <001102FF0C00>)
image/pict  application/pdf 25  nsimagetopdf
```

Drivers

- A driver is just a filter that creates printer ready data
  - The PPD for the printer specifies the proper driver

```
*cupsFilter: "application/vnd.cups-raster 0 /Library/Printers/
<vendor>/Filter/rastertovendor"
```

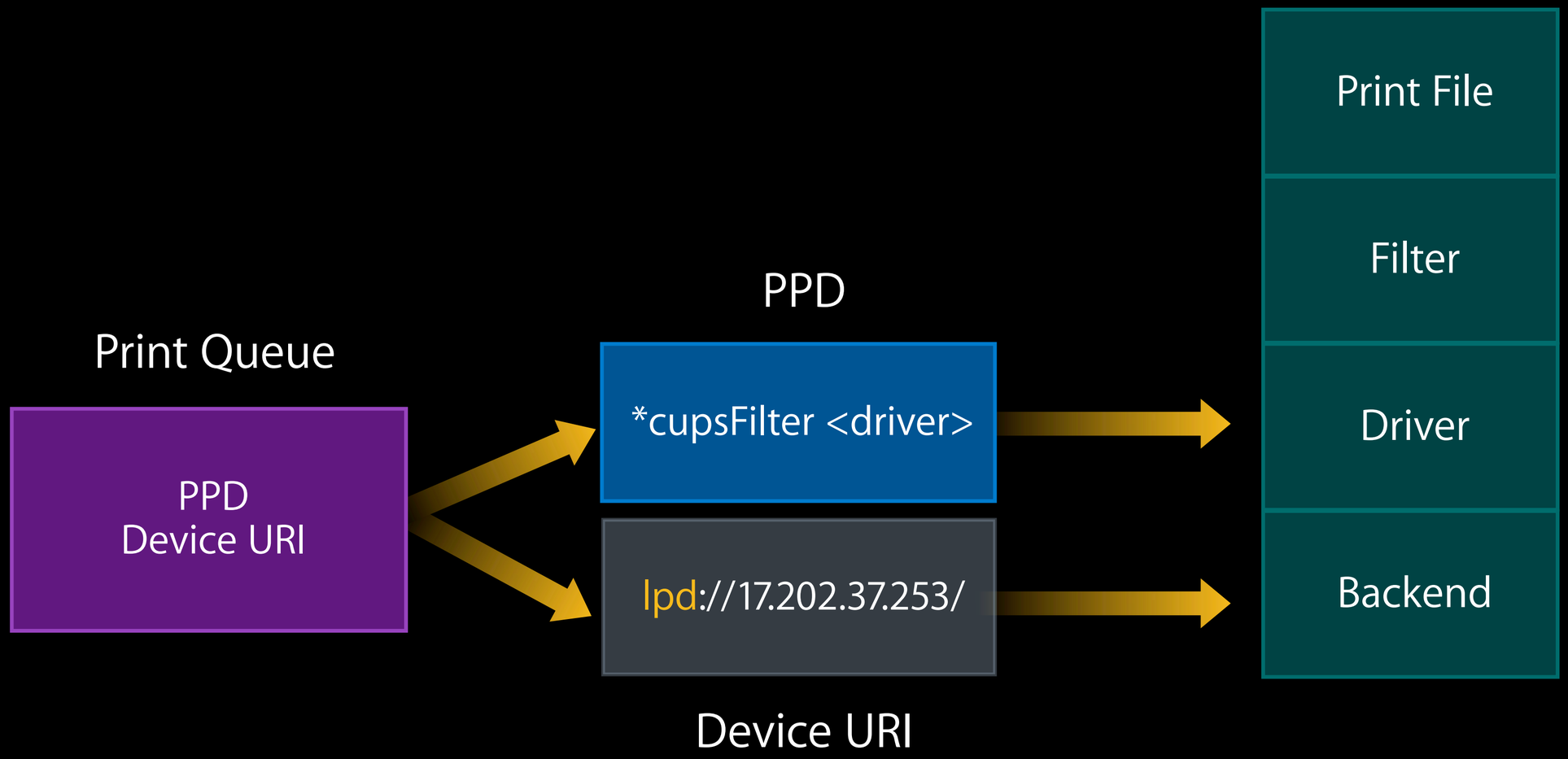
Writing a Driver

- Many drivers accept application/vnd.cups-raster
  - Supports a variety of bitmap depths and formats
  - Tiger adds 16-bits per color component raster support

Writing a Backend

- Backends send the driver ready data to the printer
- Normally take the same command line as the filters
- The backend is chosen based on the printer’s device URI

Print Queue Example



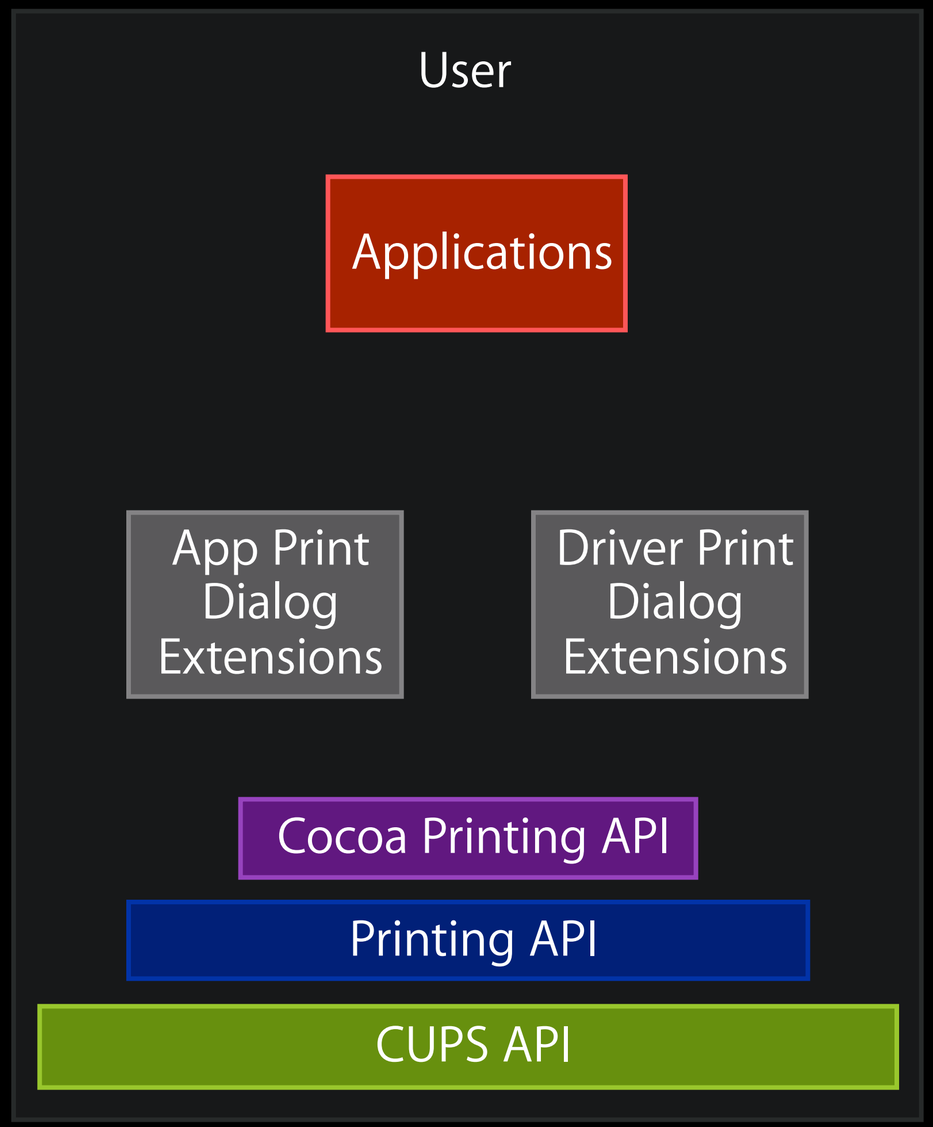
Opportunities

- Printer Vendors and others
  - Drivers
- Anybody
  - Filters and backends

Living in Two Places

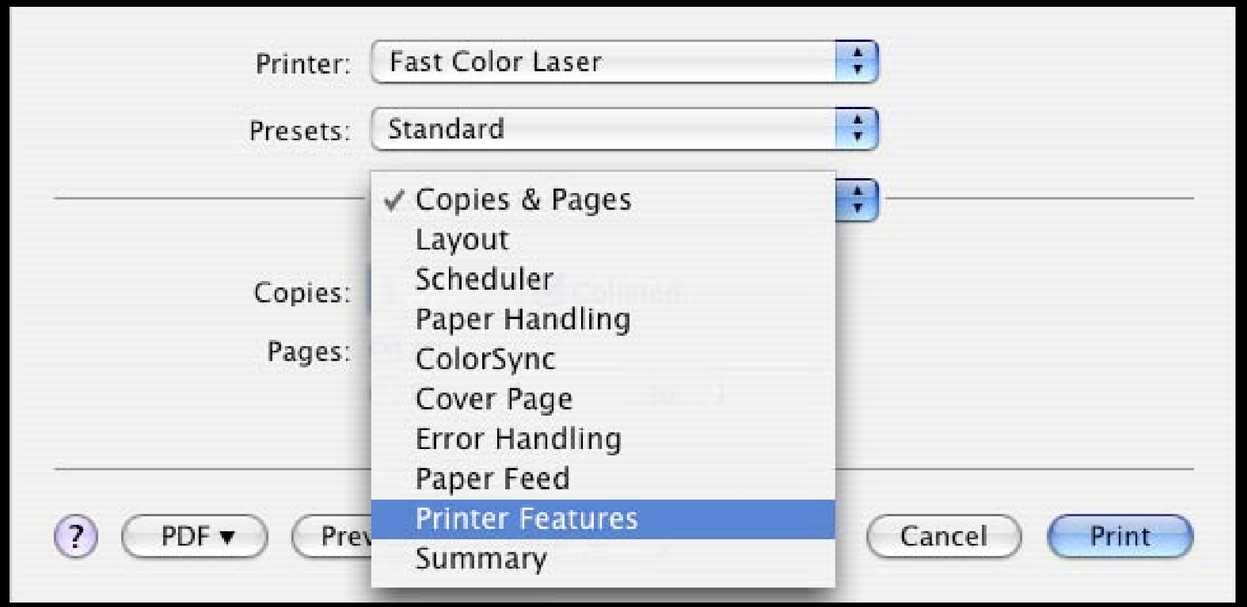
- Drivers also want to present UI in the print dialog
  - Different time
  - Different user
  - Maybe a different machine

Driver Print Dialog Extensions



Driver Print Dialog Extensions

- The print system will build UI from the PPD
  - Or the PPD can specify a print dialog extension



Print Dialog Extensions

- Run inside of the hosting application
  - Run with the user's id
  - Can crash the app

Dialog Extension Example

Add this to the PPD

```
*APDialogExtension: "/Library/Printers/<vendor>/PDEs/<bundle>"
```

Tiger Cocoa Print Dialog Extensions

- PDEPluginInterface.h
- PDEPlugIn informal protocol

```
- (NSArray*)PDEPanelsForType:(NSString*)pdeType withHostInfo:(id)host;
```

  - Returns an array of PDEPanel instances

Tiger Cocoa Print Dialog Extensions

PDEPanel informal protocol

```
- (NSView*)panelView;
- (void)willShow;
- (BOOL)shouldHide;
- (BOOL)saveValuesAndReturnError:(NSError **)error;
- (BOOL)restoreValuesAndReturnError:(NSError **)error;
```

PDEPlugInCallbackProtocol

Informal protocol used to get information from the print system

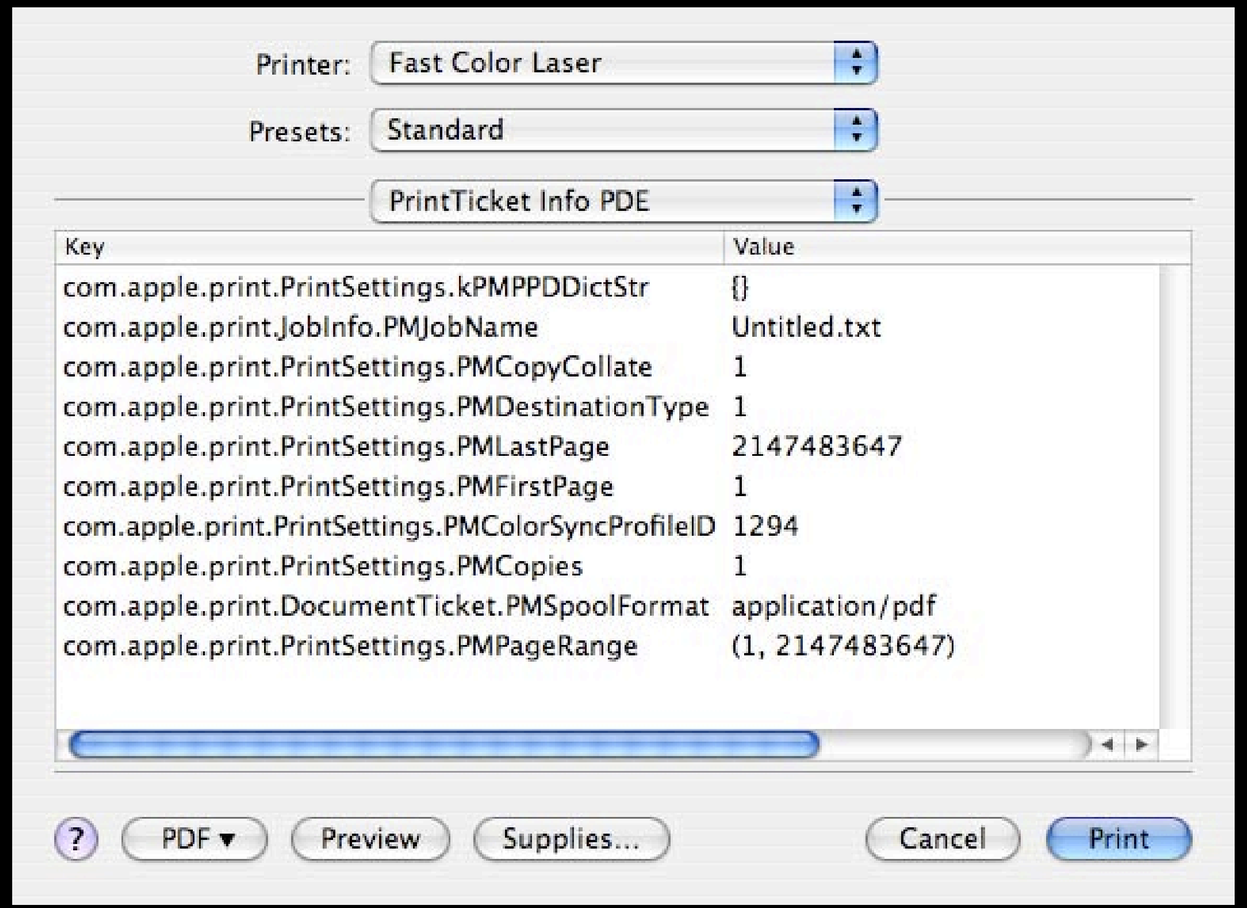
```
- (PMPrintSession) printSession;
- (PMPrintSettings) printSettings;
- (PMPageFormat) pageFormat;
- (PMPrinter) PMPrinter;
- (ppd_file_t*) ppdFile;
```

Print Settings vs. Job Tickets

Confusion between app level print settings and plugin level job tickets

- Cocoa print dialog extensions have access to both
- Print settings is preferred

Sample


Printer: Fast Color Laser
Presets: Standard
PrintTicket Info PDE
Key
Value
com.apple.print.PrintSettings.kPMPPDDictStr
{}
com.apple.print.JobInfo.PMJobName
Untitled.txt
com.apple.print.PrintSettings.PMCopyCollate
1
com.apple.print.PrintSettings.PMDestinationType
1
com.apple.print.PrintSettings.PMLastPage
2147483647
com.apple.print.PrintSettings.PMFirstPage
1
com.apple.print.PrintSettings.PMColorSyncProfileID
1294
com.apple.print.PrintSettings.PMCopies
1
com.apple.print.DocumentTicket.PMSpoolFormat
application/pdf
com.apple.print.PrintSettings.PMPageRange
(1, 2147483647)
?
PDF ▾
Preview
Supplies...
Cancel
Print

More PPD Links

- The PPD can also link
  - Icon
    - *APPrinterIconPath: <path>
  - Printer utility
    - *APPrinterUtilityPath
  - Low ink tool
    - *APPrinterLowInkTool

Low Ink Tool

If no low ink tool is supplied

- SNMP query of IP device is attempted
- To try Apple's tool

```
System/Library/Printers/Libraries/snmpInk <printer_id>
```

- To test your tool

```
defaults write com.apple.print verboseInk 'YES'
```

Low Ink Tool

- Tool is used to
  - Detect low supplies
  - Show supply levels
- Tiger has an SNMP default tool
  - It can be overridden

XML Description of Supplies

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">

<plist version="1.0">
<dict>
        <key>prtMarkerSuppliesTable</key>
        <array>
            // Array of supply dictionaries
        </array>
</dict>
</plist>
```

Supply Dictionary Keys

- Required Keys
  - prtMarkerSuppliesClass
  - prtMarkerSuppliesDescription
  - prtMarkerSuppliesMaxCapacity
  - prtMarkerSuppliesLevel

Supply Dictionary Keys

- Optional Keys
  - prtMarkerSuppliesType
  - prtMarkerSuppliesSupplyUnit
  - prtMarkerSRGBRepresentation
  - prtMarkerPartNumber

SNMP to Setup Printer

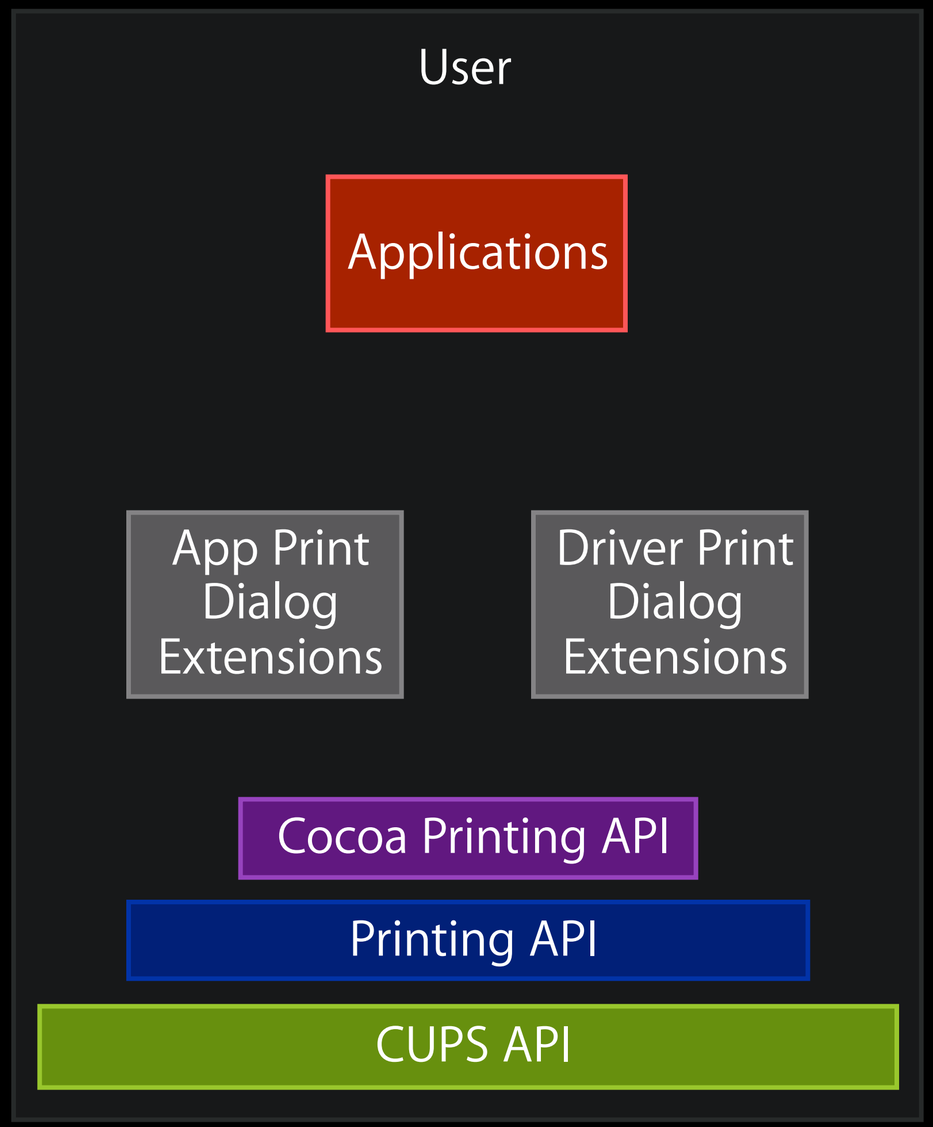
- Determining installable options
  - Relies on PPD entries to link
    - PPD options
    - SNMP values

Making SNMP Queries

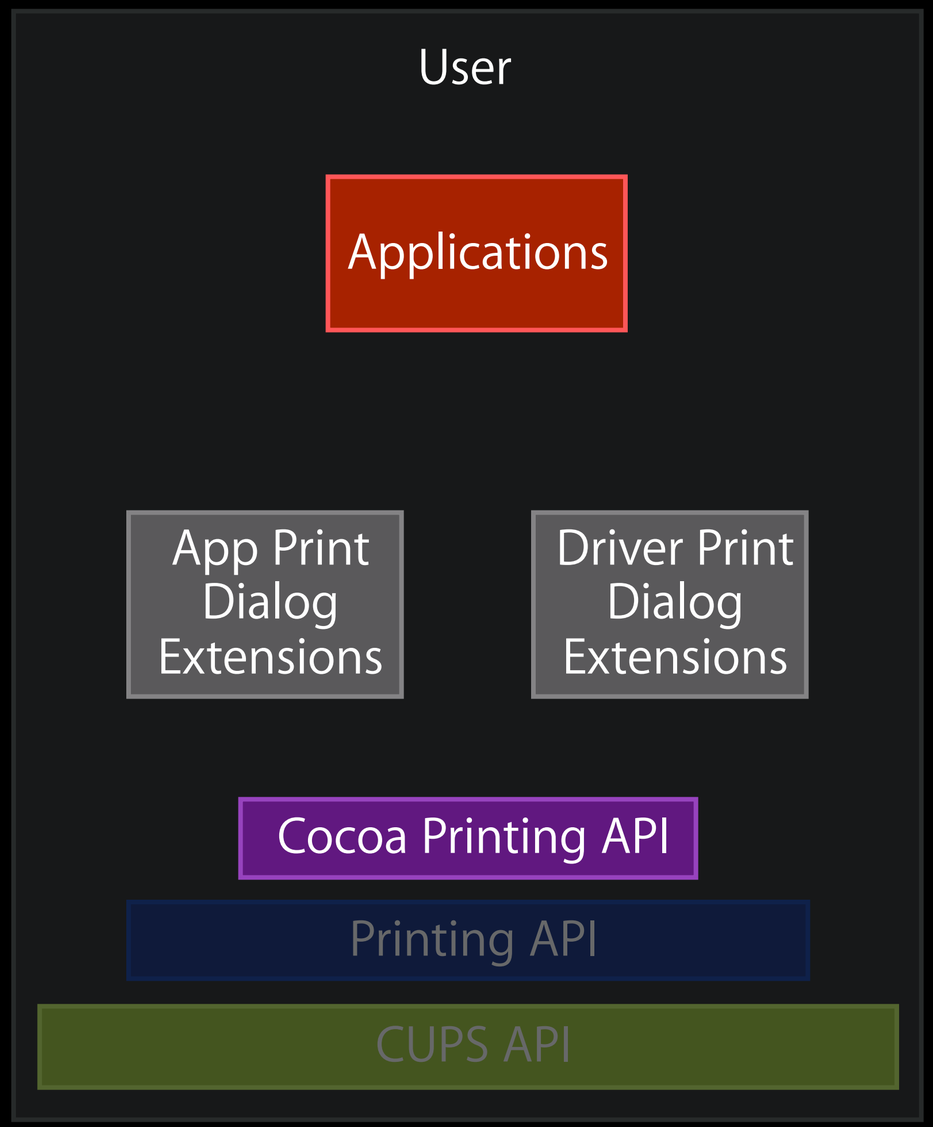
```
*OpenUI Tray3/Tray 3: Boolean
*DefaultTray3: False
*Tray3 True/Installed: "tray3"
*Tray3 False/Not Installed: "none"
*?Tray3: "<postscript to query printer>"
*End
*CloseUI: *Tray3

*?OIDTray3: ".1.3.6.1.2.1.43.8.2.1.13.1.3"
*OIDTray3 True: "installed"
```

Application Printing APIs



Application Printing APIs



Cocoa Printing APIs

For most Cocoa apps NSPrintOperation is all that is needed

```
 NSPrintOperation *op = [NSPrintOperation
         printOperationWithView:[self documentView]
         printInfo:[self printInfo]];

[op runOperationModalForWindow: [self documentWindow]
         delegate:self
         didRunSelector: @selector
               (printOpDidRun:success:contextInfo:)
         contextInfo:NULL];
```

Use Accessory Views

When a Cocoa app wants to add a panel to the print dialog

- Use an accessory view
  - [op setAccessoryView:[self printAccessoryView]];

- Some limitations
  - Only one panel can be added
  - No direct access to underlying Carbon data

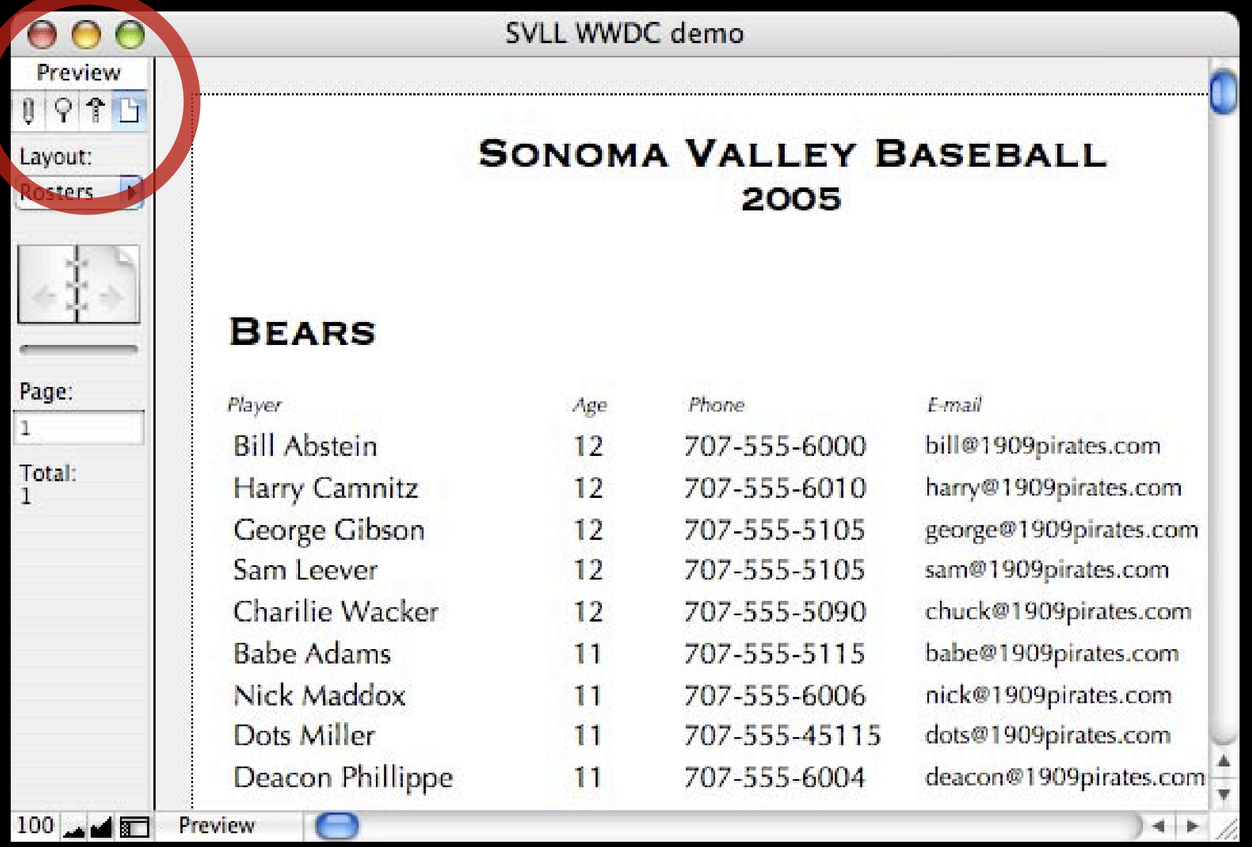
Accessory Views

- Ideally an app wouldn’t add an accessory view
  - The app has menus and can add dialogs
- Use the accessory view to provide print options, not formatting options

A Call for WYSIWYG

- Where possible use your document window to show the print view
  - Don’t do it in the print dialog
  - Let the user reformat the page and edit the data in the print view
  - If it is a good print format then it is likely a good presentation format

FileMaker


SVLL WWDC demo
Preview
Layout:
Rosters
Page:
1
Total:
1
100
Preview
Sonoma Valley Baseball
2005
Bears
Player
Age
Phone
E-mail
Bill Abstein 12 707-555-6000 bill@1909pirates.com
Harry Camnitz 12 707-555-6010 harry@1909pirates.com
George Gibson 12 707-555-5105 george@1909pirates.com
Sam Leever 12 707-555-5105 sam@1909pirates.com
Charilie Wacker 12 707-555-5090 chuck@1909pirates.com
Babe Adams 11 707-555-5115 babe@1909pirates.com
Nick Maddox 11 707-555-6006 nick@1909pirates.com
Dots Miller 11 707-555-45115 dots@1909pirates.com
Deacon Phillippe 11 707-555-6004 deacon@1909pirates.com

Other Good Examples

- Pages
- TextEdit Wrap to Page
- SOHO Labels and Envelopes
- iPhoto Book View

Bad Examples

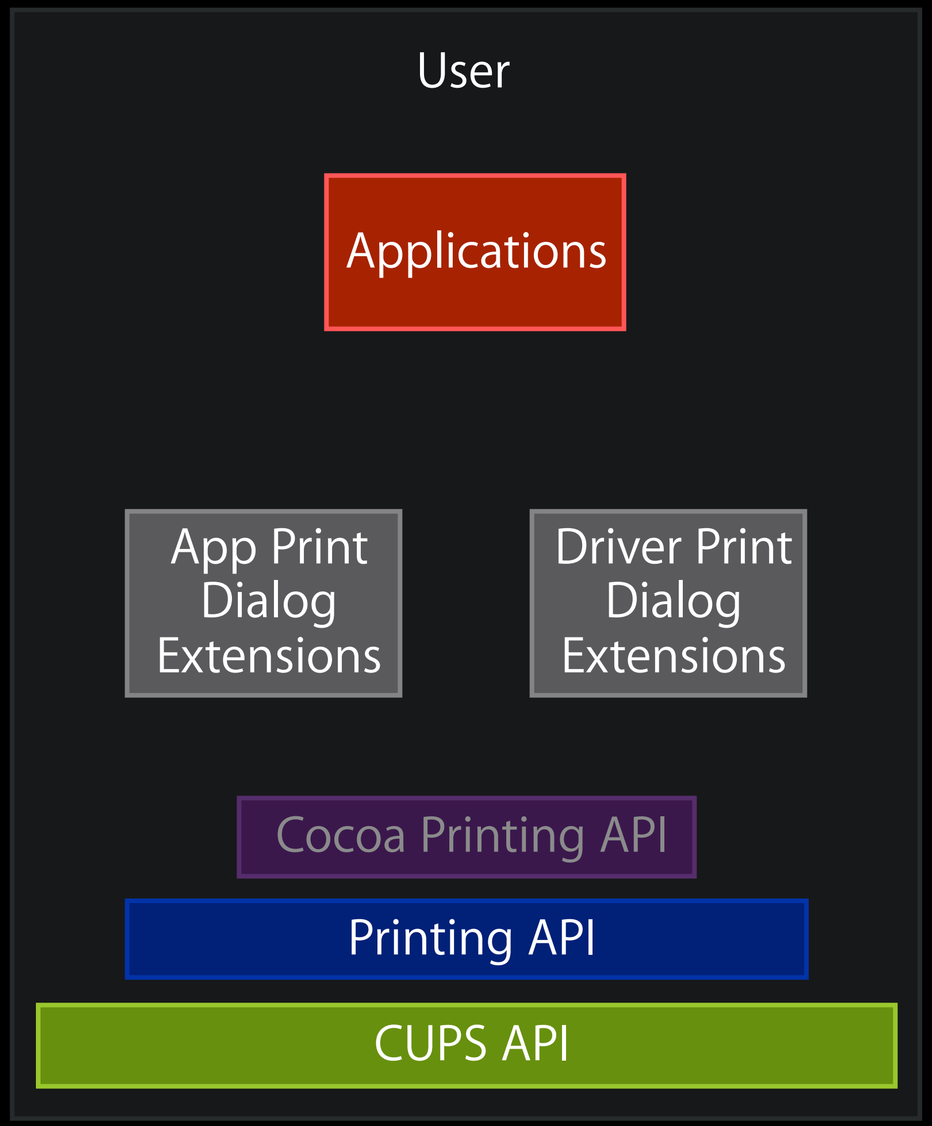
- Sadly, too many and too embarrassing to enumerate
- You may be heading the wrong way
  - If you are trying to stuff a preview into the print dialog
  - If you add formatting options that cannot be previewed in your application

WYSIWYG Returns

- Avoid putting print formatting options in the print dialog
- Avoid a separate dialog before the print dialog
- Let the user format in the main window using the full power of the app's user interface

Printing API

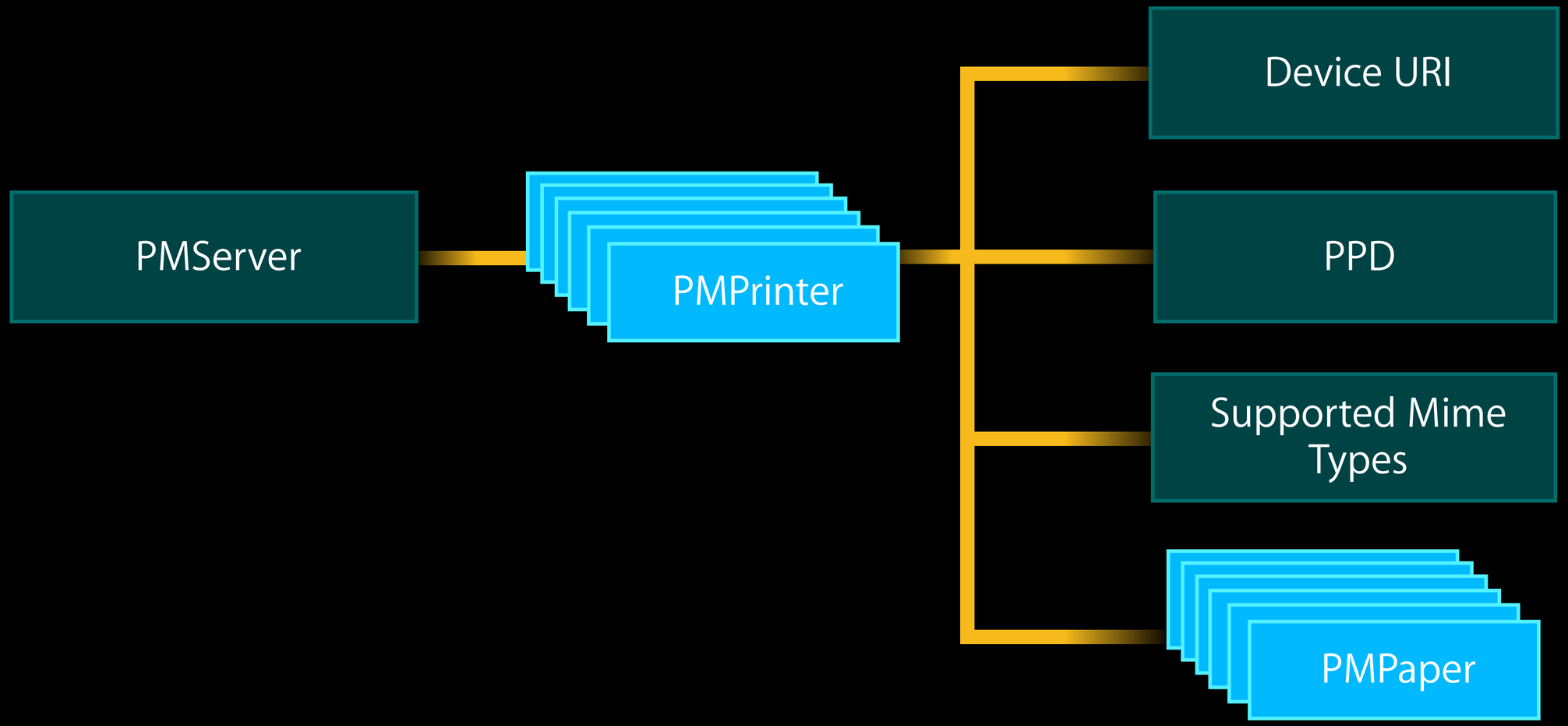
The full printing API lies below Cocoa



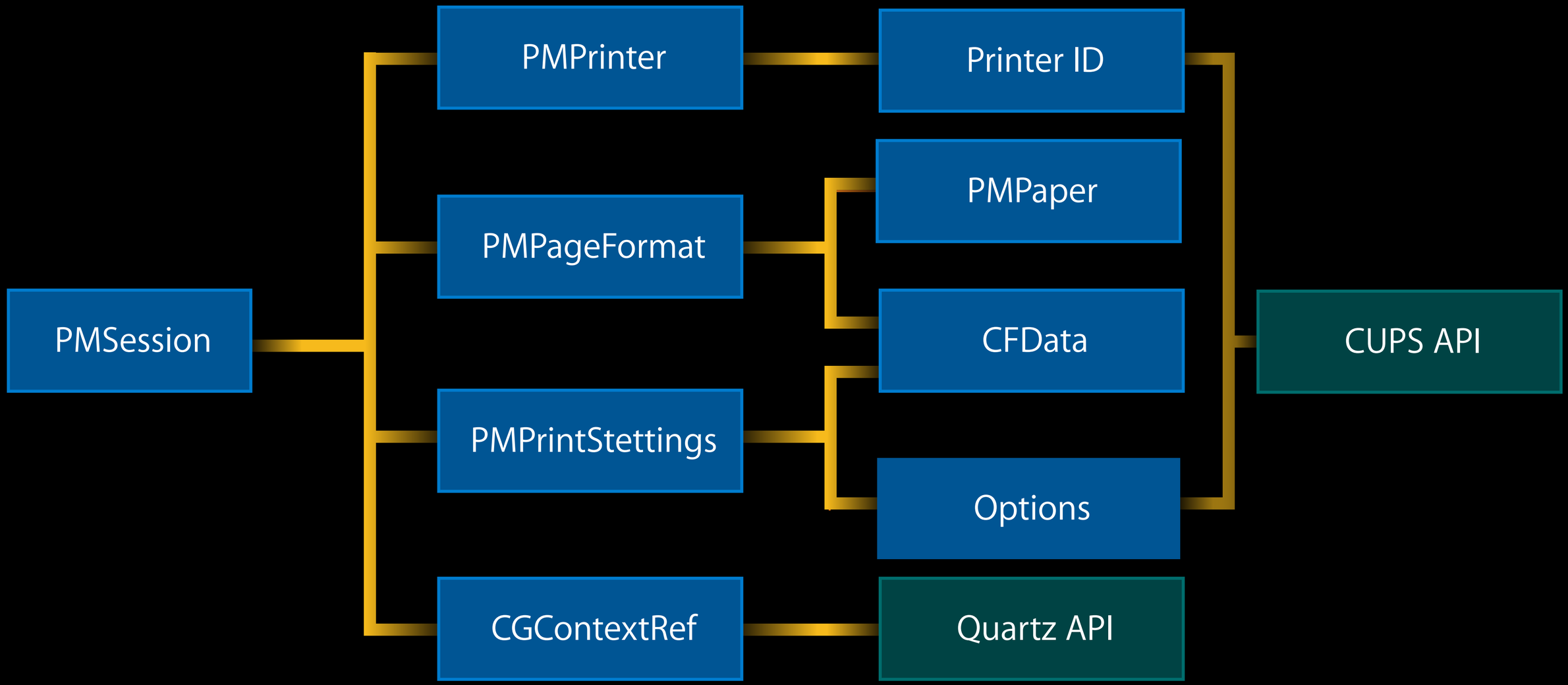
Printing API

- Use Cocoa Printing API when you can
  - It’s a better fit for Cocoa applications
- Use the Printing API directly if
  - You have a Carbon app
  - Your Cocoa app needs more control over printing

Printing Objects



Printing Objects



Printing

- Three important pieces of a job
  - Where to print it
    - PMPrinter
  - How to print it
    - PMPrintSettings
  - How it fits on paper
    - PMPageFormat

PMPrinter

http://developer.apple.com/wwdc2005/

PMPrinter

- Get the print queues for the local machine

```
OSStatus PMServerCreatePrinterList(
                PMServer server,
                CFArrayRef *printerList)
```

  - Use kPMServerLocal for the server

PMPrinter

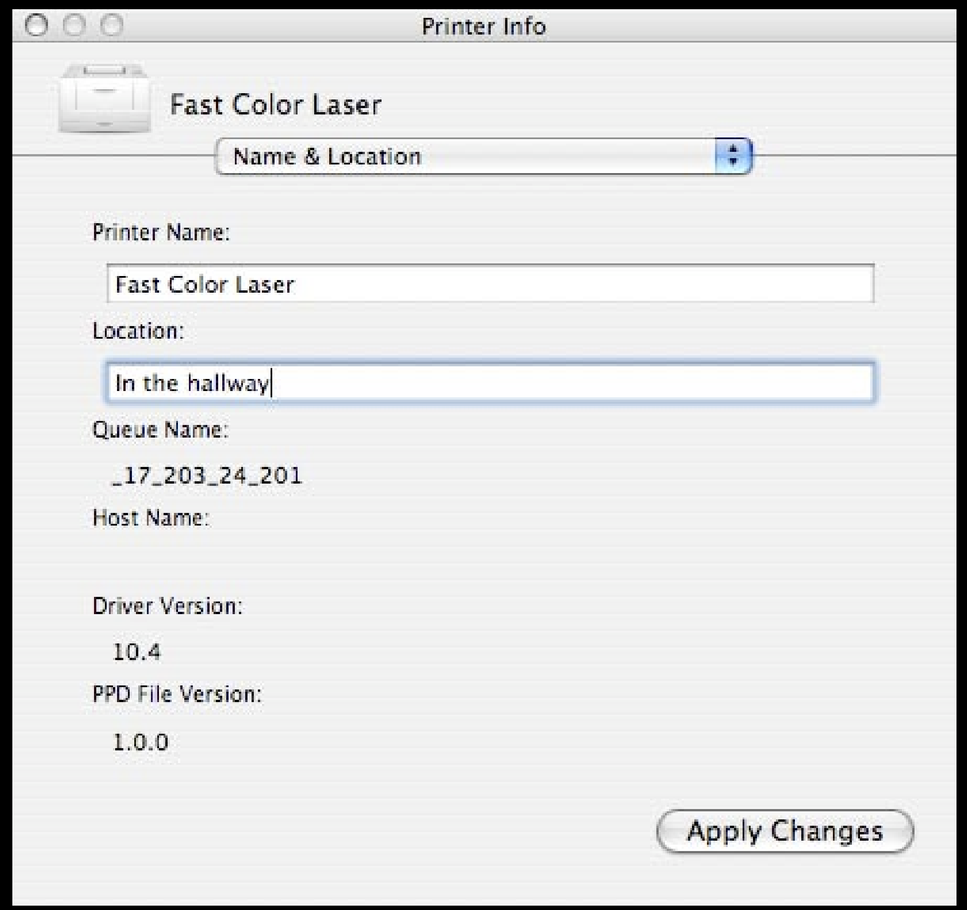
A PMPrinter has many attributes

- ID
  - The ID is unique on a given Mac
  - The ID is the link to the CUPS APIs
    - CFStringRef PMPrinterGetID(PMPrinter printer)
- Name
  - The name shown to the user

PMPrinter

- The ID and name are often different
  - The ID has a restricted character set
  - The ID is used on the command line
  - The user can rename the PMPrinter

Printer ID and Name



PMPrinter Atrributes

- Is the queue on the local Mac
  - PMPrinterIsRemote()
- The PPD for the printer
  - PMPrinterCopyTempPPDURL()
- The device URI
  - Specifies the backend to be used for printing
  - PMPrinterCopyDeviceURI()
    - PMPrinterIsDefault()
    - PMPrinterCanDuplex()
    - PMPrinterHasCustomPaperSizes

PMPrinter Attributes

- Many other PMPrinter attributes
  - PMPrinterIsDefault()
  - PMPrinterIsFavorite()
  - PMPrinterGetMakeAndModelName()
  - PMPrinterGetLocation()
  - PMPrinterCanDuplex()
  - PMPrinterHasCustomPaperSizes()

PMPrintSettings

http://developer.apple.com/wwdc2005/

PMPrintSettings

- The print dialog is the preferred way to alter print settings for a print job
  - Mixes user interface options from
    - Print system
    - Printer driver
    - Application
- Duplicating this changing mix is very hard

Flatten/Unflatten

- For Tiger there are new calls to flatten and unflatten print settings
  - Not handle-based
  - PMFlattenPrintSettingsToCFData()
  - PMFlattenPrintSettingsToURL()
  - Unflatten equivalents

PMPrintSettings Attributes

- Many specific calls to get specific attributes

```
PMGetCopies()
PMGetDuplex()
…
```

PMPrintSettings Attributes

For Tiger we have general Get/Set calls

```
OSStatus PMPrintSettingsGetValue(
         PMPrintSettings printSettings,
         CFStringRef key,
         CFTypeRef *value)

OSStatus PMPrintSettingsSetValue(
         PMPrintSettings printSettings,
         CFStringRef key,
         CFTypeRef value,
         Boolean locked)
```

PMPrintSetting to CUPS

CUPS APIs want a list of options

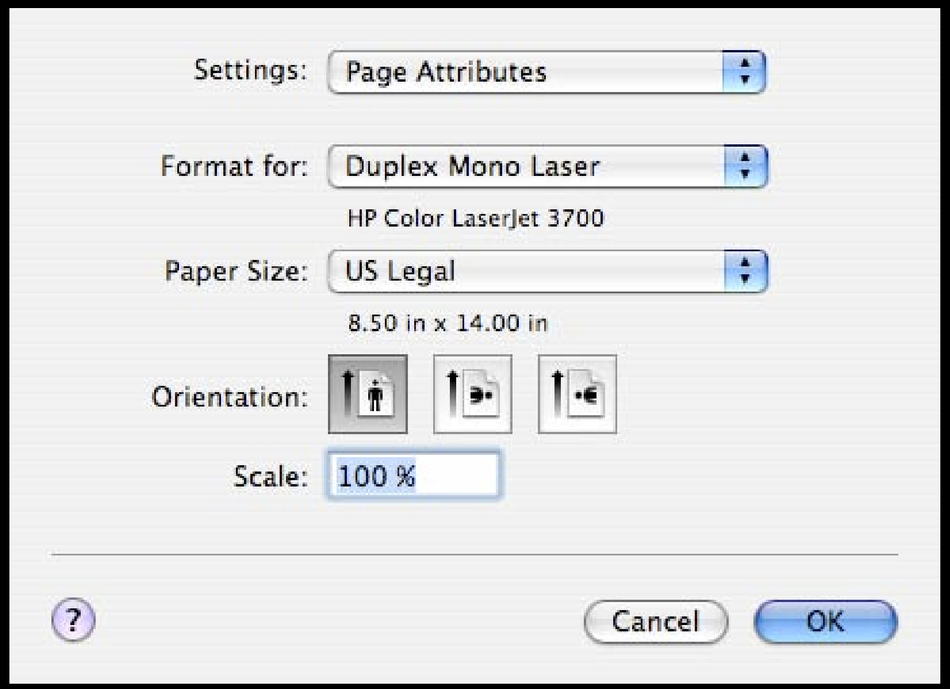
```
OSStatus PMPrintSettingsToOptions(
         PMPrintSettings settings,
         char **options)
```

PMPageFormat

http://developer.apple.com/wwdc2005/

Page Format

If at all possible use the Page Setup dialog to let the user chose a page format



PMPageFormat

- A PMPageFormat wraps
  - PMPaper
  - Orientation
  - Scaling—for now

Finding a Paper Size

- If Page Setup dialog is not appropriate
  - Get the list of PMPapers for a printer
  - Find the paper you want
  - Create a PMPageFormat from the PMPaper

PMPaper List

```
OSStatus PMPrinterGetPaperList(
         PMPrinter printer,
         CFArrayRef *paperList )
```

- Each PMPaper instance has
  - Height, width, margins
  - Use attributes to find desired paper

PMPageFormat

- Create the PMPageFormat from the PMPaper

```
OSStatus PMCreatePageFormatWithPMPaper(
         PMPageFormat* pageFormat,
         PMPaper paper )
```

- Set the orientation on the PMPageFormat
  - PMSetOrientation()

Creating a PMPaper

If you want a particular paper size regardless of printer's capabilities

```
OSStatus PMPaperCreate(
        PMPrinter printer,
        CFStringRef id,
        CFStringRef name,
        double width,
        double height,
        const PMPaperMargins *margins,
        PMPaper *paperP)
```

The Paper Menu

App Paper Popup

- If you want to create your own paper popup
- You can get information about the suggested hierarchical paper menu

```
OSStatus PMPrinterCreatePaperInfoListForMenu(
        PMPrinter printer,
        PMPageFormat pageFormat,
        CFArrayRef *paperInfoList)
```

Direct Printing

http://developer.apple.com/wwdc2005/

Direct Printing API

Normally apps will render with Quartz

- Sometimes apps have PDF or PostScript files to print
- Use the direct printing API
- Preview uses this API

Direct Printing API

Determine if your mime type is supported

```
OSStatus PMPrinterGetMimeTypes(
        PMPrinter printer,
        PMPrintSettings settings,
        CFArrayRef *mimeTypesP)
```

Direct Printing API

Spool the file to the print queue

```
OSStatus PMPrinterPrintWithFile(
        PMPrinter printer,
        PMPrintSettings settings,
        PMPageFormat format,
        CFStringRef mimeType,
        CFURLRef file)
```

Printing Without a Printer

- Getting a list of papers for a printer
- Direct Printing API

Destinations

- Print jobs don’t always go to a printer
- Destination
  - Type
  - Format
  - Location

Destination Type

- PMSessionGetDestinationType()
  - kPMDestinationPrinter
  - kPMDestinationFile
  - kPMDestinationPreview
  - kPMDestinationProcessPDF

Destination Format

PMSessionCopyDestinationFormat()

- The mime type of the output to be generated
  - PDF
  - PostScript

Destination Location

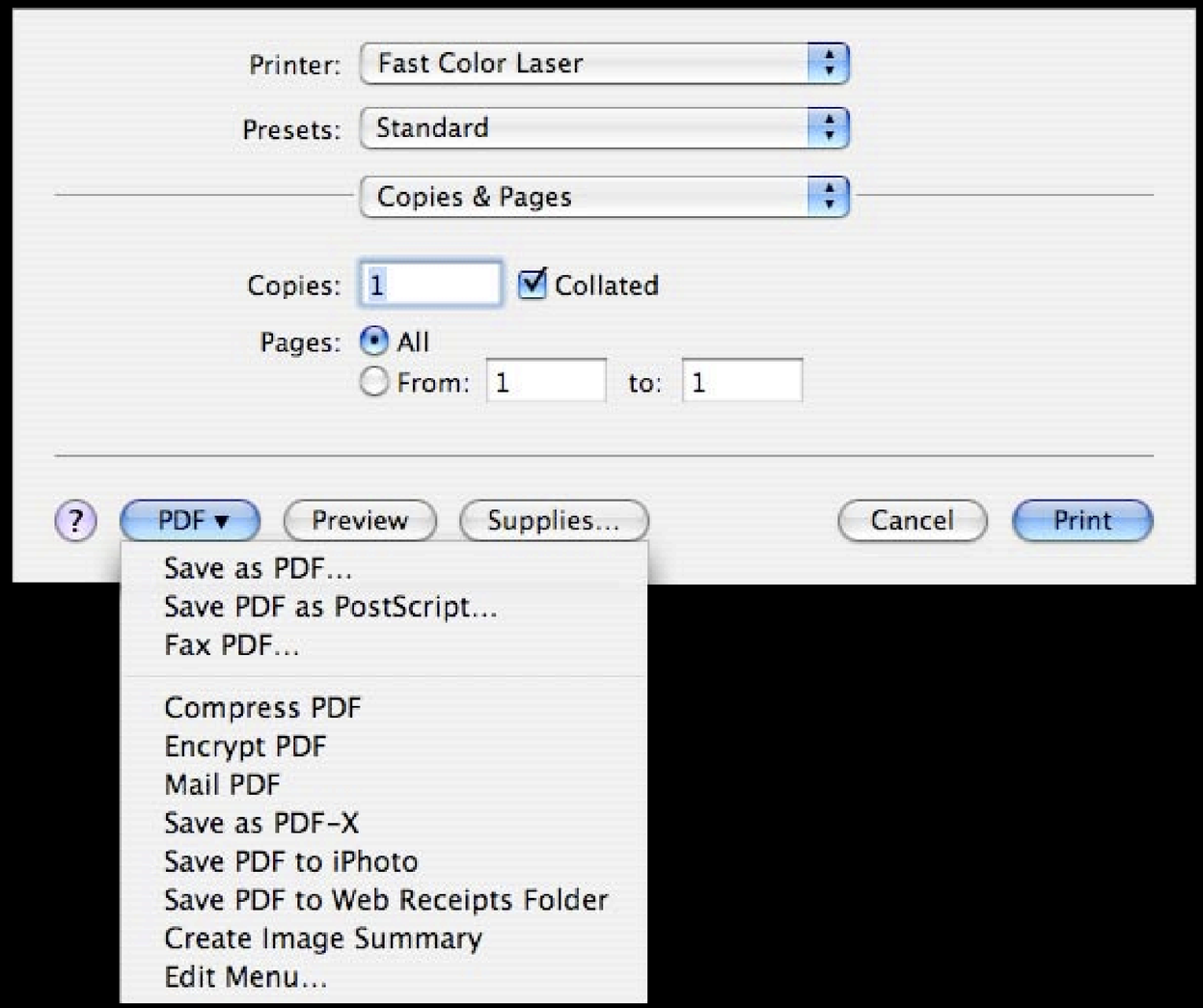
PMSessionCopyDestinationLocation()

- Used for some destination types
  - Save to file
  - Process PDF

Setting the Destination

```
OSStatus PMSessionSetDestination(
         PMPrintSession printSession,
         PMPrintSettings printSettings,
         PMDestinationType destType,
         CFStringRef destFormat,
         CFURLRef destLocation)
```

PDF Workflow



Summary

- There are a lot of places to hook and use the print system
  - Filters, drivers, backends
  - Print dialog extensions
  - Easy Cocoa printing
  - Rich Printing API
  - Underlying CUPS API

Related Session

Printing	Lab Friday 9:00AM

Q&A

http://developer.apple.com/wwdc2005/

WWDC2005

Worldwide Developers Conference